
Self-supervised Graph-level Representation Learning with Local and Global Structure

Minghao Xu¹ Hang Wang¹ Bingbing Ni¹ Hongyu Guo² Jian Tang^{3,4,5}

Abstract

This paper studies unsupervised/self-supervised whole-graph representation learning, which is critical in many tasks such as molecule properties prediction in drug and material discovery. Existing methods mainly focus on preserving the local similarity structure between different graph instances but fail to discover the global semantic structure of the entire data set. In this paper, we propose a unified framework called **Local-instance** and **Global-semantic Learning** (*GraphLoG*) for self-supervised whole-graph representation learning. Specifically, besides preserving the local similarities, GraphLoG introduces the *hierarchical prototypes* to capture the global semantic clusters. An efficient online expectation-maximization (EM) algorithm is further developed for learning the model. We evaluate GraphLoG by pre-training it on massive unlabeled graphs followed by fine-tuning on downstream tasks. Extensive experiments on both chemical and biological benchmark data sets demonstrate the effectiveness of the proposed approach.

1. Introduction

Learning informative representations of whole graphs is a fundamental problem in a variety of domains and tasks, such as molecule properties prediction in drug and material discovery (Gilmer et al., 2017; Wu et al., 2018), protein function forecast in biological networks (Alvarez & Yan, 2012; Jiang et al., 2017), and predicting the properties of circuits in circuit design (Zhang et al., 2019). Recently, Graph Neural Networks (GNNs) have attracted a surge of interest and showed the effectiveness of learning graph representations (Hamilton et al., 2017; Xu et al., 2019; Shi et al.,

2020). These methods are usually trained in a supervised fashion, which requires a large number of labeled data. Nevertheless, in many scientific domains, labeled data are very limited and expensive to obtain. Therefore, it is becoming increasingly important to learn the representations of graphs in an unsupervised or self-supervised fashion.

Self-supervised learning has recently achieved profound success for both natural language processing, *e.g.* GPT (Radford et al., 2018) and BERT (Devlin et al., 2019), and image understanding, *e.g.* MoCo (He et al., 2019) and SimCLR (Chen et al., 2020). However, how to effectively learn the representations of graphs in a self-supervised way is still an open problem. Intuitively, a desirable graph representation should be able to preserve the *local-instance structure*, so that similar graphs are embedded close to each other and dissimilar ones stay far apart. In addition, the representations of the entire set of graphs should also reflect the *global-semantic structure* of the data, so that the graphs with similar semantic properties are compactly embedded, which is able to benefit various downstream tasks such as graph classification or regression. Such global structure can be effectively captured by semantic clusters (Caron et al., 2018; Ji et al., 2019), which can be further organized hierarchically (Li et al., 2020).

There are some recent works that learn graph representation in a self-supervised manner, such as local-global mutual information maximization (Velickovic et al., 2019; Sun et al., 2019), structural-similarity/context prediction (Navarin et al., 2018; Hu et al., 2019; You et al., 2020b), contrastive learning (Hassani & Ahmadi, 2020; Qiu et al., 2020; You et al., 2020a) and meta-learning (Lu et al., 2021). However, all these methods are able to model only the local structure between different graph instances but fail to discover the global-semantic structure. To address this shortcoming, we are seeking for an approach that is sufficient to model both the local and global structure of a set of graphs.

To attain this goal, we propose a **Local-instance** and **Global-semantic Learning** (*GraphLoG*) framework for self-supervised graph representation learning. Specifically, for preserving the local similarity between various graph instances, we seek to align the embeddings of correlated graphs/subgraphs by discriminating the correlated

¹Shanghai Jiao Tong University ²National Research Council Canada ³Mila - Quebec AI Institute ⁴CIFAR AI Research Chair ⁵HEC Montréal. Correspondence to: Minghao Xu <xuminghao118@sjtu.edu.cn>, Jian Tang <jian.tang@hec.ca>.

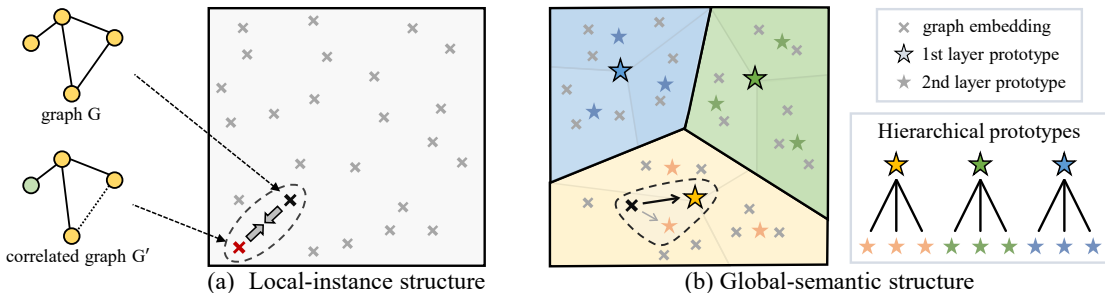


Figure 1. **Illustration of GraphLoG.** (a) Correlated graphs are constrained to be adjacently embedded to pursue the local-instance structure of the data. (b) Hierarchical prototypes are employed to discover and refine the global-semantic structure of the data.

graph/subgraph pairs from the negative pairs. In this locally smooth latent space, we further introduce the additional model parameters, *hierarchical prototypes*¹, to depict the latent distribution of a graph data set in a hierarchical way. For model learning, we propose to maximize the data likelihood with respect to both the GNN parameters and hierarchical prototypes via an online expectation-maximization (EM) algorithm. Given a mini-batch of graphs sampled from the data distribution, in the E-step, we infer the embeddings of these graphs with a GNN and sample the latent variable of each graph (*i.e.* the prototypes associated to each graph) from the posterior distribution defined by current model. In the M-step, we aim to maximize the expectation of complete-data likelihood with respect to the current model by optimizing with a mini-batch-induced objective function. Therefore, in this iterative EM process, the global-semantic structure of the data can be gradually discovered and refined. The whole model is pre-trained with a large number of unlabeled graphs, and then fine-tuned and evaluated on some downstream tasks containing scarce labeled graphs.

To verify the effectiveness of the GraphLoG framework, we apply our method to both the chemistry and biology domains. Through pre-training on massive unlabeled molecular graphs (or protein ego-networks) using the proposed local and global objectives, the existing GNN models are able to achieve superior performance on the downstream molecular property (or biological function) prediction benchmarks. In particular, the Graph Isomorphism Network (GIN) (Xu et al., 2019) pre-trained by the proposed method outperforms the previous self-supervised graph representation learning approaches on six of eight downstream tasks of chemistry domain, and it achieves a 2.1% performance gain in terms of average ROC-AUC on eight downstream tasks. In addition, the analytical experiments further illustrate the benefits of global structure learning through conducting ablation studies and visualizing the embedding distributions on a set of graphs.

¹Hierarchical prototypes are representative cluster embeddings organized as a set of trees.

2. Problem Definition and Preliminaries

2.1. Problem Definition

An ideal representation should preserve the local structure among various data instances. More specifically, we define it as follows:

Definition 1 (Local-instance Structure). The local structure aims to preserve the pairwise similarity between different instances after mapping from the high-dimensional input space to the low-dimensional latent space (Roweis & Saul, 2000; Belkin & Niyogi, 2002). For a pair of similar graphs/subgraphs, \mathcal{G} and \mathcal{G}' , their embeddings are expected to be nearby in the latent space, as illustrated in Fig. 1(a), while the dissimilar pairs should be mapped to far apart.

The pursuit of local-instance structure alone is usually insufficient to capture the semantics underlying the entire data set. It is therefore important to discover the global-semantic structure of the data, which is concretely defined as follows:

Definition 2 (Global-semantic Structure). A real-world data set can usually be organized as various semantic clusters (Furnas et al., 2017; Ji et al., 2019), especially in a hierarchical way for graph-structured data (Ashburner et al., 2000; Chen et al., 2012b). After mapping to the latent space, the embeddings of a set of graphs are expected to form some global structures reflecting the clustering patterns of the original data. A graphical illustration can be seen in Fig. 1(b).

Problem Definition. The problem of *self-supervised graph representation learning* considers a set of unlabeled graphs $\mathbf{G} = \{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_M\}$, and we aim at learning a low-dimensional vector $h_{\mathcal{G}_m} \in \mathbb{R}^d$ for each graph $\mathcal{G}_m \in \mathbf{G}$ under the guidance of the data itself. In specific, we expect the graph embeddings $\mathbf{H} = \{h_{\mathcal{G}_1}, h_{\mathcal{G}_2}, \dots, h_{\mathcal{G}_M}\}$ follow both the local-instance and global-semantic structure.

2.2. Preliminaries

Graph Neural Networks (GNNs). A typical graph can be represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, X_{\mathcal{V}}, X_{\mathcal{E}})$, where \mathcal{V} is a set of

nodes, \mathcal{E} denotes the edge set, $X_{\mathcal{V}} = \{X_v | v \in \mathcal{V}\}$ stands for the attributes of all nodes, and $X_{\mathcal{E}} = \{X_{uv} | (u, v) \in \mathcal{E}\}$ represents the edge attributes. A GNN aims to learn an embedding vector $h_v \in \mathbb{R}^{\delta}$ for each node $v \in \mathcal{V}$ and also a vector $h_{\mathcal{G}} \in \mathbb{R}^{\delta}$ for the entire graph \mathcal{G} . For an L -layer GNN, a neighborhood aggregation scheme is performed to capture the L -hop information surrounding each node. As suggested in Gilmer et al. (2017), the l -th layer of a GNN can be formalized as follows:

$$h_v^{(l)} = f_U^{(l)}(h_v^{(l-1)}, f_M^{(l)}(\{(h_u^{(l-1)}, h_u^{(l-1)}, X_{uv}) : u \in \mathcal{N}(v)\})), \quad (1)$$

where $\mathcal{N}(v)$ is the neighborhood set of v , $h_v^{(l)}$ denotes the representation of node v at the l -th layer, $h_v^{(0)}$ is initialized as the node attribute X_v , and $f_M^{(l)}$ and $f_U^{(l)}$ stand for the message passing and update function at the l -th layer, respectively. Since h_v summarizes the information of a subgraph centered around node v , we will refer to h_v as *subgraph embedding* to underscore this point. The entire graph’s embedding can be derived as below:

$$h_{\mathcal{G}} = f_R(\{h_v | v \in \mathcal{V}\}), \quad (2)$$

where f_R is a permutation-invariant readout function, *e.g.* mean pooling or more complex graph-level pooling function (Ying et al., 2018; Zhang et al., 2018).

General EM Algorithm. The basic objective of EM algorithm (Dempster et al., 1977; Krishnan et al., 1997) is to find the maximum likelihood solution for a model containing latent variables. In such a problem, we denote the set of all observed data as \mathbf{X} , the set of all latent variables as \mathbf{Z} and the set of all model parameters as θ .

In the E-step, using the data \mathbf{X} and the model parameters θ_{t-1} estimated by the last EM cycle, the posterior distribution of latent variables is derived as $p(\mathbf{Z} | \mathbf{X}, \theta_{t-1})$ which can also be regarded as the responsibility that a specific set of latent variables are taken for explaining the observations.

In the M-step, employing the posterior distribution given by the E-step, the expectation of complete-data log-likelihood, denoted as $Q(\theta)$, is evaluated for the general model parameters θ as follows:

$$Q(\theta) = \mathbb{E}_{p(\mathbf{Z} | \mathbf{X}, \theta_{t-1})}[\log p(\mathbf{X}, \mathbf{Z} | \theta)]. \quad (3)$$

The model parameters are updated to maximize this expectation in the M-step, which outputs:

$$\theta_t = \arg \max_{\theta} Q(\theta). \quad (4)$$

Each cycle of EM can increase the complete-data likelihood expected by the current model, and, considering the whole progress, the EM algorithm has been demonstrated to be capable of maximizing the marginal likelihood function $p(\mathbf{X} | \theta)$ (Hathaway, 1986; Neal & Hinton, 1998).

3. GraphLoG: Self-supervised Graph-level Representation Learning with Local and Global Structure

In this section, we introduce our approach called Local-instance and Global-semantic Learning (GraphLoG) for self-supervised graph representation learning. The main purpose of GraphLoG is to discover and refine both the local and global structures of graph embeddings in the latent space, such that we can learn useful graph representations for the downstream task like graph classification. Specifically, GraphLoG constructs a locally smooth latent space by aligning the embeddings of correlated graphs/subgraphs. On such basis, the global structures of graph embeddings are modeled by hierarchical prototypes, and the data likelihood is maximized via an online EM algorithm. Next, we elucidate the GraphLoG framework in detail.

3.1. Learning Local-instance Structure of Graph Representations

Following the existing methods for dimensionality reduction (Tenenbaum et al., 2000; Roweis & Saul, 2000; Belkin & Niyogi, 2002), the goal of local-structure learning is to preserve the local similarity of the data before and after mapping to a low-dimensional latent space. In specific, it is expected that similar graphs or subgraphs are embedded close to each other, and dissimilar ones are mapped to far apart. Using a similarity measurement defined in the latent space, we formulate this problem as maximizing the similarity of correlated graph/subgraph pairs while minimizing that of negative pairs.

In specific, given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, X_{\mathcal{V}}, X_{\mathcal{E}})$ sampled from the data distribution $P_{\mathcal{G}}$, we obtain its correlated counterpart $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', X_{\mathcal{V}'}, X_{\mathcal{E}'})$ through randomly masking a part of node/edge attributes in the graph (Hu et al., 2019) (see appendix for the detailed scheme). In addition, for a subgraph \mathcal{G}_v constituted by node v and its L -hop neighborhoods in graph \mathcal{G} , we regard the corresponding subgraph \mathcal{G}'_v in graph \mathcal{G}' as its correlated counterpart. Through applying an L -layer GNN model GNN_{θ} (θ stands for GNN’s parameters) upon graph \mathcal{G} and \mathcal{G}' , the graph and subgraph embeddings are derived as follows:

$$(h_{\mathcal{V}}, h_{\mathcal{G}}) = \text{GNN}_{\theta}(\mathcal{G}), \quad (h_{\mathcal{V}'}, h_{\mathcal{G}'}) = \text{GNN}_{\theta}(\mathcal{G}'), \quad (5)$$

where $h_{\mathcal{V}} = \{h_{\mathcal{G}_v} | v \in \mathcal{V}\}$ and $h_{\mathcal{V}'} = \{h_{\mathcal{G}'_v} | v \in \mathcal{V}'\}$ represent the set of subgraph embeddings within graph \mathcal{G} and \mathcal{G}' , respectively.

In this phase, the objective of learning is to enhance the similarity of correlated graph/subgraph pairs while diminish that of negative pairs. Using a specific similarity measure (*e.g.* cosine similarity $s(x, y) = x^{\top} y / \|x\| \|y\|$ in our practice), we seeks to optimize the following two objective functions

for graph and subgraph, respectively:

$$\mathcal{L}_{\text{graph}} = -\mathbb{E}_{(\mathcal{G}_+, \mathcal{G}'_+) \sim p(\mathcal{G}, \mathcal{G}'), (\mathcal{G}_-, \mathcal{G}'_-) \sim p_n(\mathcal{G}, \mathcal{G}')} [s(\mathcal{G}_+, \mathcal{G}'_+) - s(\mathcal{G}_-, \mathcal{G}'_-)], \quad (6)$$

$$\mathcal{L}_{\text{sub}} = -\mathbb{E}_{(\mathcal{G}_u, \mathcal{G}'_u) \sim p(\mathcal{G}_v, \mathcal{G}'_v), (\mathcal{G}_v, \mathcal{G}'_v) \sim p_n(\mathcal{G}_v, \mathcal{G}'_v)} [s(\mathcal{G}_u, \mathcal{G}'_u) - s(\mathcal{G}_v, \mathcal{G}'_v)], \quad (7)$$

where $p_n(\mathcal{G}, \mathcal{G}')$ and $p_n(\mathcal{G}_v, \mathcal{G}'_v)$ denote the noise distribution from which negative pairs are sampled. In practice, for a correlated graph pair $(\mathcal{G}, \mathcal{G}')$ or correlated subgraph pair $(\mathcal{G}_v, \mathcal{G}'_v)$, we substitute \mathcal{G} (\mathcal{G}_v) randomly with another graph from the data set (a subgraph centered around another node in the same graph) to construct negative pairs.

For learning the local-instance structure of graph representations, we aim to minimize both objective functions (Eqs. 6 and 7) with respect to the parameters of GNN:

$$\min_{\theta} \mathcal{L}_{\text{local}}, \quad (8)$$

$$\mathcal{L}_{\text{local}} = \mathcal{L}_{\text{graph}} + \mathcal{L}_{\text{sub}}. \quad (9)$$

3.2. Learning Global-semantic Structure of Graph Representations

It is worth noticing that the graphs in a data set may possess hierarchical semantic information. For example, drugs (*i.e.* molecular graphs) are represented by a five-level hierarchy in the Anatomical Therapeutic Chemical (ATC) classification system (Chen et al., 2012b). After mapping to the latent space, the embeddings of all graphs in the data set are also expected to form some global structures corresponding to the hierarchical semantic structures of the original data.

However, for the lack of explicit semantic labels in the self-supervised graph representation learning, such global structure cannot be attained via label-induced supervision. To tackle this limitation, we introduce an additional set of model parameters, *hierarchical prototypes*, to represent the feature clusters in the latent space in a hierarchical way. They are formally defined as $\mathbf{C} = \{c_i^l\}_{i=1}^{M_l}$ ($l = 1, 2, \dots, L_p$), where $c_i^l \in \mathbb{R}^\delta$ stands for the i -th prototype at the l -th layer, L_p is the depth of hierarchical prototypes, and M_l denotes the number of prototypes at the l -th layer. These prototypes are structured as a set of trees (Fig. 1(b)), in which each node corresponds to a prototype, and, except for the leaf nodes, each prototype possesses a set of child nodes, denoted as $\mathbb{C}(c_i^l)$ ($1 \leq i \leq M_l$, $l = 1, 2, \dots, L_p - 1$).

The goal of global-semantic learning is to encourage the graphs to be compactly embedded around corresponding prototypes and, at the same time, refine hierarchical prototypes to better represent the data. We formalize this problem as optimizing a latent variable model. Specifically, for the

observed data set $\mathbf{G} = \{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_M\}$, we consider a latent variable set, *i.e.* the prototype assignments of all graphs $\mathbf{Z} = \{z_{\mathcal{G}_1}, z_{\mathcal{G}_2}, \dots, z_{\mathcal{G}_M}\}$ ($z_{\mathcal{G}_m}$ is a set of prototypes that best represent \mathcal{G}_m in the latent space). The model parameters in this problem are the GNN parameters θ and hierarchical prototypes \mathbf{C} . Since the corresponding latent variable of each graph is not given, it is hard to directly maximize the complete-data likelihood function $p(\mathbf{G}, \mathbf{Z}|\theta, \mathbf{C})$. Therefore, we seek to maximize the expectation of complete-data likelihood through the EM algorithm.

The vanilla EM algorithm (Dempster et al., 1977; Krishnan et al., 1997) requires a full pass through the data set before each parameter update, which is computationally inefficient when the size of data set is large like in our case. Therefore, we consider an online EM variant (Sato & Ishii, 2000; Cappé & Moulines, 2009; Liang & Klein, 2009) which operates on mini-batches of data. This approach is based on the *i.i.d.* assumption of the data set, where both the complete-data likelihood and the posterior probability of latent variables can be factorized over each observed-latent variable pair:

$$p(\mathbf{G}, \mathbf{Z}|\theta, \mathbf{C}) = \prod_{m=1}^M p(\mathcal{G}_m, z_{\mathcal{G}_m}|\theta, \mathbf{C}), \quad (10)$$

$$p(\mathbf{Z}|\mathbf{G}, \theta, \mathbf{C}) = \prod_{m=1}^M p(z_{\mathcal{G}_m}|\mathcal{G}_m, \theta, \mathbf{C}). \quad (11)$$

First, we introduce the initialization scheme of model parameters.

Initialization of model parameters. Before triggering the global structure exploration, we first pre-train the GNN by minimizing $\mathcal{L}_{\text{local}}$ and employ the derived GNN model as initialization, which establishes a locally smooth latent space. After that, we utilize this pre-trained GNN model to extract the embeddings of all graphs in the data set, and the K-means clustering is applied upon these graph embeddings to initialize the bottom layer prototypes (*i.e.* $\{c_i^{L_p}\}_{i=1}^{M_{L_p}}$) with the output cluster centers. The prototypes of upper layers are initialized by iteratively applying K-means clustering to the prototypes of the layer below. For each time of clustering, we discard the output cluster centers assigned with less than two samples to avoid trivial solutions (Bach & Harchaoui, 2007; Caron et al., 2018).

Next, we state the details of the E-step and M-step applied in our method.

E-step. In this step, we first randomly sample a mini-batch of graphs $\tilde{\mathbf{G}} = \{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_N\}$ (N denotes the batch size) from the data set \mathbf{G} , and $\tilde{\mathbf{Z}} = \{z_{\mathcal{G}_n}\}_{n=1}^N$ stands for the latent variables corresponding to these sampled graphs. Each latent variable $z_{\mathcal{G}_n} = \{z_{\mathcal{G}_n}^1, z_{\mathcal{G}_n}^2, \dots, z_{\mathcal{G}_n}^{L_p}\}$ is a chain of prototypes from top layer to bottom layer that best represent graph \mathcal{G}_n in the latent space, and it holds that $z_{\mathcal{G}_n}^{l+1}$ is

the child node of $z_{\mathcal{G}_n}^l$ in the corresponding tree structure, *i.e.* $z_{\mathcal{G}_n}^{l+1} \in \mathbb{C}(z_{\mathcal{G}_n}^l)$ ($l = 1, 2, \dots, L_p - 1$). The posterior distribution of $\tilde{\mathbf{Z}}$ can be evaluated in a factorized way because of the *i.i.d.* assumption:

$$p(\tilde{\mathbf{Z}}|\tilde{\mathbf{G}}, \theta_{t-1}, \mathbf{C}_{t-1}) = \prod_{n=1}^N p(z_{\mathcal{G}_n}|\mathcal{G}_n, \theta_{t-1}, \mathbf{C}_{t-1}), \quad (12)$$

where θ_{t-1} and \mathbf{C}_{t-1} are the model parameters from the last EM cycle. Directly evaluating each posterior distribution $p(z_{\mathcal{G}_n}|\mathcal{G}_n, \theta_{t-1}, \mathbf{C}_{t-1})$ is nontrivial, which requires to traverse all the possible chains in hierarchical prototype. Instead, we adopt the idea of stochastic EM algorithm (Celeux & Govaert, 1992; Nielsen et al., 2000) and draw a sample $\hat{z}_{\mathcal{G}_n} \sim p(z_{\mathcal{G}_n}|\mathcal{G}_n, \theta_{t-1}, \mathbf{C}_{t-1})$ for Monte Carlo estimation. In specific, we sequentially sample a prototype from each layer in a top-down manner, and all the sampled prototypes form a connected chain from top layer to bottom layer in hierarchical prototypes. Formally, we first sample a prototype from top layer according to a categorical distribution over all the top layer prototypes, *i.e.* $\hat{z}_{\mathcal{G}_n}^1 \sim \text{Cat}(z_{\mathcal{G}_n}^1|\{\alpha_i\}_{i=1}^{M_1})$ ($\alpha_i = \text{softmax}(s(c_i^1, h_{\mathcal{G}_n}))$), where s denotes the cosine similarity measurement; for the sampling at layer l ($l \geq 2$), we draw a prototype from that layer based on a categorical distribution over the child nodes of prototype $\hat{z}_{\mathcal{G}_n}^{l-1}$ sampled from the layer above, *i.e.* $\hat{z}_{\mathcal{G}_n}^l \sim \text{Cat}(z_{\mathcal{G}_n}^l|\{\alpha_c\})$ ($\alpha_c = \text{softmax}(s(c, h_{\mathcal{G}_n}))$, $\forall c \in \mathbb{C}(\hat{z}_{\mathcal{G}_n}^{l-1})$), such that we sample a latent variable $\hat{z}_{\mathcal{G}_n} = \{\hat{z}_{\mathcal{G}_n}^1, \hat{z}_{\mathcal{G}_n}^2, \dots, \hat{z}_{\mathcal{G}_n}^{L_p}\}$ which is a connected chain in hierarchical prototypes. Using the latent variables inferred as above, we seek to maximize the expectation of complete-data log-likelihood in the M-step.

M-step. In this step, we aim at maximizing the expected complete-data log-likelihood with respect to the posterior distribution of latent variables, which is defined as follows:

$$Q(\theta, \mathbf{C}) = \mathbb{E}_{p(\mathbf{Z}|\mathbf{G}, \theta_{t-1}, \mathbf{C}_{t-1})}[\log p(\mathbf{G}, \mathbf{Z}|\theta, \mathbf{C})]. \quad (13)$$

This expectation needs the computation over all data points, which cannot be attained in the online setting. As a substitute, we propose to maximize the expected log-likelihood on mini-batch $\tilde{\mathbf{G}}$, which can be estimated using the latent variables $\tilde{\mathbf{Z}}_{est} = \{\hat{z}_{\mathcal{G}_n}\}_{n=1}^N$ sampled in the E-step:

$$\begin{aligned} \tilde{Q}(\theta, \mathbf{C}) &= \mathbb{E}_{p(\tilde{\mathbf{Z}}|\tilde{\mathbf{G}}, \theta_{t-1}, \mathbf{C}_{t-1})}[\log p(\tilde{\mathbf{G}}, \tilde{\mathbf{Z}}|\theta, \mathbf{C})] \\ &\approx \log p(\tilde{\mathbf{G}}, \tilde{\mathbf{Z}}_{est}|\theta, \mathbf{C}) \\ &= \sum_{n=1}^N \log p(\mathcal{G}_n, \hat{z}_{\mathcal{G}_n}|\theta, \mathbf{C}). \end{aligned} \quad (14)$$

We would like to point out that $\tilde{Q}(\theta, \mathbf{C})$ is a decent proxy for $Q(\theta, \mathbf{C})$, where a proportional relation approximately holds between them (see appendix for the proof):

$$\tilde{Q}(\theta, \mathbf{C}) \approx \frac{N}{M} Q(\theta, \mathbf{C}). \quad (15)$$

We further scale $\tilde{Q}(\theta, \mathbf{C})$ with the batch size to derive the log-likelihood function $\mathcal{L}(\theta, \mathbf{C})$ that is more stable in terms of computation:

$$\mathcal{L}(\theta, \mathbf{C}) = \frac{1}{N} \tilde{Q}(\theta, \mathbf{C}). \quad (16)$$

To estimate $\mathcal{L}(\theta, \mathbf{C})$, we need to define the joint likelihood of a graph \mathcal{G} and a latent variable $z_{\mathcal{G}}$, which is represented with an energy-based formulation in our method:

$$p(\mathcal{G}, z_{\mathcal{G}}|\theta, \mathbf{C}) = \frac{1}{Z(\theta, \mathbf{C})} \exp(f(h_{\mathcal{G}}, z_{\mathcal{G}})), \quad (17)$$

where $Z(\theta, \mathbf{C})$ denotes the partition function. We formalize the negative energy function f by measuring the similarities between graph embedding $h_{\mathcal{G}}$ and the prototypes in $z_{\mathcal{G}}$ and also measuring the similarities between the prototypes in $z_{\mathcal{G}}$ that are from consecutive layers:

$$f(h_{\mathcal{G}}, z_{\mathcal{G}}) = \sum_{l=1}^{L_p} s(h_{\mathcal{G}}, z_{\mathcal{G}}^l) + \sum_{l=1}^{L_p-1} s(z_{\mathcal{G}}^l, z_{\mathcal{G}}^{l+1}). \quad (18)$$

Intuitively, f evaluates how well a latent variable $z_{\mathcal{G}}$ represents graph \mathcal{G} in the latent space, and it also measures the affinity between the consecutive prototypes along a chain from top layer to bottom layer in hierarchical prototypes.

It is nontrivial to optimize with $p(\mathcal{G}, z_{\mathcal{G}}|\theta, \mathbf{C})$ due to the intractable partition function. Inspired by Noise-Contrastive Estimation (NCE) (Gutmann & Hyvärinen, 2010; 2012), we seek to optimize with the unnormalized likelihoods, *i.e.* $\tilde{p}(\mathcal{G}, z_{\mathcal{G}}|\theta, \mathbf{C}) = \exp(f(h_{\mathcal{G}}, z_{\mathcal{G}}))$, by contrasting the positive observed-latent variable pair with the negative pairs sampled from some noise distribution, which defines an objective function that well approximates $\mathcal{L}(\theta, \mathbf{C})$:

$$\begin{aligned} \mathcal{L}_{\text{global}} &= -\mathbb{E}_{(\mathcal{G}^+, z_{\mathcal{G}}^+) \sim p(\mathcal{G}, z_{\mathcal{G}})} \left\{ \log \tilde{p}(\mathcal{G}^+, z_{\mathcal{G}}^+|\theta, \mathbf{C}) \right. \\ &\quad \left. - \mathbb{E}_{(\mathcal{G}^-, z_{\mathcal{G}}^-) \sim p_n(\mathcal{G}, z_{\mathcal{G}})} [\log \tilde{p}(\mathcal{G}^-, z_{\mathcal{G}}^-|\theta, \mathbf{C})] \right\}, \end{aligned} \quad (19)$$

where $p_n(\mathcal{G}, z_{\mathcal{G}})$ is the noise distribution. In practice, we compute the outer expectation with all the positive pairs in the mini-batch, *i.e.* $(\mathcal{G}_n, \hat{z}_{\mathcal{G}_n})$ ($1 \leq n \leq N$), and, for computing the inner expectation, we construct L_p negative pairs for the positive pair $(\mathcal{G}_n, \hat{z}_{\mathcal{G}_n})$ by fixing the graph \mathcal{G}_n and randomly substituting one of L_p prototypes in $\hat{z}_{\mathcal{G}_n}$ with another prototype at the same layer each time. For global-semantic learning, we aim to minimize the global objective function $\mathcal{L}_{\text{global}}$ with respect to both the GNN parameter θ and hierarchical prototypes \mathbf{C} :

$$\min_{\theta, \mathbf{C}} \mathcal{L}_{\text{global}}. \quad (20)$$

In general, the proposed online EM algorithm seeks to maximize the joint likelihood $p(\mathbf{G}, \mathbf{Z}|\theta, \mathbf{C})$ governed by model parameters θ and \mathbf{C} . For a step further, we propose the following proposition that this algorithm can indeed maximize the marginal likelihood function $p(\mathbf{G}|\theta, \mathbf{C})$.

Algorithm 1 Optimization Algorithm of GraphLoG.

Input: Unlabeled graph data set \mathbf{G} , the number of learning steps T .
Output: Pre-trained GNN model GNN_{θ_T} .
Pre-train GNN with local objective function (Eq. 9).
Initialize model parameters θ_0 and \mathbf{C}_0 .
for $t = 1$ **to** T **do**
 Sample a mini-batch $\tilde{\mathbf{G}}$ from \mathbf{G} .
 \diamond *E-step*:
 Sample latent variables $\tilde{\mathbf{Z}}_{est}$ with $\text{GNN}_{\theta_{t-1}}$ and \mathbf{C}_{t-1} .
 \diamond *M-step*:
 Update model parameters:
 $\theta_t \leftarrow \theta_{t-1} - \nabla_{\theta}(\mathcal{L}_{\text{local}} + \mathcal{L}_{\text{global}})$,
 $\mathbf{C}_t \leftarrow \mathbf{C}_{t-1} - \nabla_{\mathbf{C}}(\mathcal{L}_{\text{local}} + \mathcal{L}_{\text{global}})$.
end for

Proposition 1. For each EM cycle, the model parameters θ and \mathbf{C} are updated in such a way that increases the marginal likelihood function $p(\mathbf{G}|\theta, \mathbf{C})$, unless a local maximum is reached on the mini-batch log-likelihood function $\tilde{Q}(\theta, \mathbf{C})$.

The proof of Proposition 1 is provided in the appendix.

3.3. Model Optimization and Downstream Application

The GraphLoG framework seeks to learn the graph representations preserving both the local-instance and global-semantic structure on an unlabeled graph data set \mathbf{G} . For model optimization under this framework, we first pre-train the GNN by minimizing the local objective function $\mathcal{L}_{\text{local}}$ and initialize the model parameters with the pre-trained GNN. After that, for each learning step, we sample a mini-batch $\tilde{\mathbf{G}}$ from the data set and conduct an EM cycle. In the E-step, the latent variables corresponding to the mini-batch, *i.e.* $\tilde{\mathbf{Z}}_{est}$, are sampled from the posterior distribution defined by the current model. In the M-step, model parameters are updated to maximize the expected log-likelihood on mini-batch $\tilde{\mathbf{G}}$. Also, we add the local objective function to the optimization of M-step, which guarantees the local smoothness when pursuing the global-semantic structure and performs well in practice. We summarize the optimization algorithm in Alg. 1.

After the self-supervised pre-training on massive unlabeled graphs, the derived GNN model can be applied to various downstream tasks for producing effective embedding vectors of different graphs. For example, we can first pre-train a GNN model with GraphLoG on a large number of unlabeled molecules (*i.e.* molecular graphs). After that, for a downstream task of molecular property prediction where a small set of labeled molecules are available, we can learn a linear classifier upon the pre-trained GNN model to perform the specific graph classification task.

4. Related Work

Graph Neural Networks (GNNs). Recently, following the efforts of learning graph representations via optimizing random walk (Perozzi et al., 2014; Tang et al., 2015; Grover & Leskovec, 2016; Narayanan et al., 2017) or matrix factorization (Cao et al., 2015; Wang et al., 2016) objectives, GNNs explicitly derive proximity-preserved feature vectors in a neighborhood aggregation way. As suggested in Gilmer et al. (2017), the forward pass of most GNNs can be depicted in two phases, *Message Passing* and *Readout* phase, and various works (Duvenaud et al., 2015; Kipf & Welling, 2017; Hamilton et al., 2017; Velickovic et al., 2018; Ying et al., 2018; Zhang et al., 2018; Xu et al., 2019) sought to improve the effectiveness of these two phases. Unlike these methods which are mainly trained in a supervised fashion, our approach aims for self-supervised learning for GNNs.

Self-supervised Learning for GNNs. There are some recent works that explored self-supervised graph representation learning with GNNs. García-Durán & Niepert (2017) learned graph representations by embedding propagation, and Velickovic et al. (2019), Sun et al. (2019) achieved this goal through mutual information maximization. Also, some self-supervised tasks, *e.g.* edge prediction (Kipf & Welling, 2016), context prediction (Hu et al., 2019; Rong et al., 2020), graph partitioning (You et al., 2020b), edge/attribute generation (Hu et al., 2020) and contrastive learning (Hassani & Ahmadi, 2020; Qiu et al., 2020; You et al., 2020a), have been designed to acquire knowledge from unlabeled graphs. Nevertheless, all these methods are only able to model the local relations between different graph instances. The proposed framework seeks to discover both the local-instance and global-semantic structure of a set of graphs.

Self-supervised Semantic Learning. Clustering-based methods (Xie et al., 2016; Yang et al., 2016; 2017; Caron et al., 2018; Ji et al., 2019; Li et al., 2020) are commonly used to learn the semantic information of the data in a self-supervised fashion. Among which, DeepCluster (Caron et al., 2018) proved the strong transferability of the visual representations learnt by clustering prediction to various downstream visual tasks. Prototypical Contrastive Learning (Li et al., 2020) proved its superiority over the instance-level contrastive learning approaches. These methods are mainly developed for images but not for graph-structured data. Furthermore, the hierarchical semantic structure of the data has been less explored in previous works.

5. Experiments

In this section, we evaluate the performance of GraphLoG on both the chemistry and biology domains using the procedure of pre-training followed by fine-tuning. Also, analytical studies are conducted to verify the effectiveness of local and global structure learning.

Table 1. Test ROC-AUC (%) on downstream molecular property prediction benchmarks.

Methods	BBBP	Tox21	ToxCast	SIDER	ClinTox	MUV	HIV	BACE	Avg
Random	65.8 ± 4.5	74.0 ± 0.8	63.4 ± 0.6	57.3 ± 1.6	58.0 ± 4.4	71.8 ± 2.5	75.3 ± 1.9	70.1 ± 5.4	67.0
EdgePred (2016)	67.3 ± 2.4	76.0 ± 0.6	64.1 ± 0.6	60.4 ± 0.7	64.1 ± 3.7	74.1 ± 2.1	76.3 ± 1.0	79.9 ± 0.9	70.3
InfoGraph (2019)	68.2 ± 0.7	75.5 ± 0.6	63.1 ± 0.3	59.4 ± 1.0	70.5 ± 1.8	75.6 ± 1.2	77.6 ± 0.4	78.9 ± 1.1	71.1
AttrMasking (2019)	64.3 ± 2.8	76.7 ± 0.4	64.2 ± 0.5	61.0 ± 0.7	71.8 ± 4.1	74.7 ± 1.4	77.2 ± 1.1	79.3 ± 1.6	71.1
ContextPred (2019)	68.0 ± 2.0	75.7 ± 0.7	63.9 ± 0.6	60.9 ± 0.6	65.9 ± 3.8	75.8 ± 1.7	77.3 ± 1.0	79.6 ± 1.2	70.9
GraphPartition (2020b)	70.3 ± 0.7	75.2 ± 0.4	63.2 ± 0.3	61.0 ± 0.8	64.2 ± 0.5	75.4 ± 1.7	77.1 ± 0.7	79.6 ± 1.8	70.8
GraphCL (2020a)	69.5 ± 0.5	75.4 ± 0.9	63.8 ± 0.4	60.8 ± 0.7	70.1 ± 1.9	74.5 ± 1.3	77.6 ± 0.9	78.2 ± 1.2	71.3
GraphLoG (ours)	72.5 ± 0.8	75.7 ± 0.5	63.5 ± 0.7	61.2 ± 1.1	76.7 ± 3.3	76.0 ± 1.1	77.8 ± 0.8	83.5 ± 1.2	73.4

Table 2. Test ROC-AUC (%) on downstream biological function prediction benchmark.

Methods	ROC-AUC (%)
Random	64.8 ± 1.0
EdgePred (Kipf & Welling, 2016)	70.5 ± 0.7
InfoGraph (Sun et al., 2019)	70.7 ± 0.5
AttrMasking (Hu et al., 2019)	70.5 ± 0.5
ContextPred (Hu et al., 2019)	69.9 ± 0.3
GraphPartition (You et al., 2020b)	71.0 ± 0.2
GraphCL (You et al., 2020a)	71.2 ± 0.6
GraphLoG (ours)	72.9 ± 0.7

5.1. Experimental Setup

Pre-training details. Following Hu et al. (2019), we adopt a five-layer Graph Isomorphism Network (GIN) (Xu et al., 2019) with 300-dimensional hidden units and a mean pooling readout function for performance comparisons (Secs. 5.2 and 5.3). We use an Adam optimizer (Kingma & Ba, 2015) (learning rate: 1×10^{-3}) to pre-train the GNN with $\mathcal{L}_{\text{local}}$ for one epoch and then train the whole model with both $\mathcal{L}_{\text{local}}$ and $\mathcal{L}_{\text{global}}$ for 10 epochs. For each time of K-means clustering in the initialization of hierarchical prototypes, we adopt 50 initial cluster centers. Unless otherwise specified, the batch size N is set as 512, and the hierarchical prototypes’ depth L_p is set as 3. These hyperparameters are selected by the grid search on the validation sets of four downstream molecule data sets (*i.e.* BBBP, SIDER, ClinTox and BACE), and their sensitivity is analyzed in Sec. 5.4.

Fine-tuning details. For fine-tuning on a downstream task, a linear classifier is appended upon the pre-trained GNN, and an Adam optimizer (learning rate: 1×10^{-3} , fine-tuning batch size: 32) is employed to train the model for 100 epochs. We utilize a learning rate scheduler with fix step size which multiplies the learning rate by 0.3 every 30 epochs. All the reported results are averaged over five independent runs under the same configuration, where five random seeds are used for these runs.

Performance comparison. For the experiments on both chemistry and biology domains, we compare the proposed method with existing self-supervised graph representation learning algorithms (*i.e.* EdgePred (Kipf & Welling, 2016), InfoGraph (Sun et al., 2019), AttrMasking (Hu et al., 2019), ContextPred (Hu et al., 2019), GraphPartition (You et al., 2020b) and GraphCL (You et al., 2020a)) to verify its effec-

Table 3. Test ROC-AUC (%) of different methods under four GNN architectures. (All results are reported on biology domain.)

Methods	GCN	GraphSAGE	GAT	GIN
Random	63.2 ± 1.0	65.7 ± 1.2	68.2 ± 1.1	64.8 ± 1.0
EdgePred (2016)	68.0 ± 0.9	67.8 ± 0.7	67.9 ± 1.3	70.5 ± 0.7
AttrMasking (2019)	68.3 ± 0.8	69.2 ± 0.6	67.3 ± 0.8	70.5 ± 0.5
ContextPred (2019)	67.6 ± 0.3	69.6 ± 0.6	66.9 ± 1.2	69.9 ± 0.3
GraphCL (2020a)	69.1 ± 0.9	70.2 ± 0.4	68.4 ± 1.2	71.2 ± 0.6
GraphLoG (ours)	71.2 ± 0.6	70.8 ± 0.8	69.5 ± 1.0	72.9 ± 0.7

tiveness. We report the results of EdgePred, AttrMasking and ContextPred from Hu et al. (2019) and examine the performance of InfoGraph, GraphPartition and GraphCL based on the released source code.

5.2. Experiments on Chemistry Domain

Data sets. For fair comparison, we use the same data sets as in Hu et al. (2019). In specific, a subset of ZINC15 database (Sterling & Irwin, 2015) with 2 million unlabeled molecules is employed for self-supervised pre-training. Eight binary classification data sets in MoleculeNet (Wu et al., 2018) serve as downstream tasks, where the scaffold split scheme (Chen et al., 2012a) is used for data set split.

Results. In Tab. 1, we report the performance of the proposed GraphLoG method compared with other works, where ‘Random’ denotes the GIN model with random initialization. Among all self-supervised learning strategies, our approach achieves the best performance on six of eight tasks, and a 2.1% performance gain is obtained in terms of average ROC-AUC. We deem that this improvement over previous works is mainly from the global structure modeling in GraphLoG, which is not included in existing methods.

5.3. Experiments on Biology Domain

Data sets. For biology domain, following Hu et al. (2019), 395K unlabeled protein ego-networks are utilized for self-supervised pre-training. The downstream task is to predict 40 fine-grained biological functions of 8 species.

Results. Tab. 2 reports the test ROC-AUC of various self-supervised learning techniques. It can be observed that the proposed GraphLoG method outperforms existing ap-

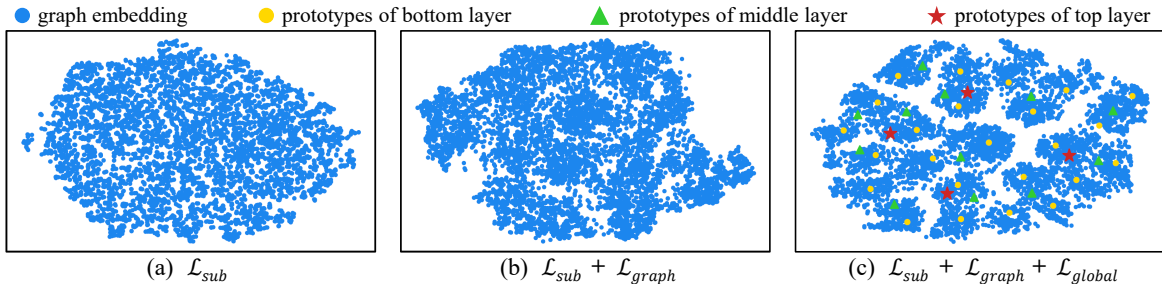


Figure 2. The t-SNE visualization on ZINC15 database (*i.e.* the pre-training data set for chemistry domain).

Table 4. Ablation study for different objective functions on downstream biological function prediction benchmark.

\mathcal{L}_{sub}	\mathcal{L}_{graph}	\mathcal{L}_{global}	ROC-AUC (%)
✓			70.1 ± 0.6
	✓		71.0 ± 0.3
		✓	71.5 ± 0.5
✓	✓		71.3 ± 0.7
✓		✓	71.9 ± 0.8
	✓	✓	72.2 ± 0.4
✓	✓	✓	72.9 ± 0.7

proaches with a clear margin, *i.e.* a 1.7% performance gain. This result illustrates that the proposed approach is able to learn effective graph representations that benefit the downstream task involving fine-grained classification.

In Tab. 3, we further compare GraphLoG with four existing methods under four GNN architectures (*i.e.* GCN (Kipf & Welling, 2017), GraphSAGE (Hamilton et al., 2017), GAT (Velickovic et al., 2018) and GIN (Xu et al., 2019)). We can observe that GraphLoG outperforms the existing approaches on all configurations, and, compared to EdgePred, AttrMasking and ContextPred, it avoids the performance decrease relative to random initialization baseline on GAT.

5.4. Analysis

Effect of different objective functions. In Tab. 4, we analyze the effect of three objective functions on the biology domain, and we continue using the GIN depicted in Sec. 5.1 in this experiment. When each objective function is individually applied (1st, 2nd and 3rd row), the one for global-semantic learning performs best, which probably benefits from its exploration of the semantic structure of the data. Through simultaneously applying different objective functions, the full model (last row) achieves the best performance, which illustrates that the learning of local and global structure are complementary to each other.

Sensitivity of hierarchical prototypes’ depth L_p . In this part, we discuss the selection of parameter L_p which controls the number of discovered semantic hierarchies. In Fig. 3(a), we plot model’s performance under different L_p values. It can be observed that deeper hierarchical prototypes (*i.e.* $L_p \geq 3$) achieve stable performance gain compared to the shallow ones (*i.e.* $L_p \leq 2$).

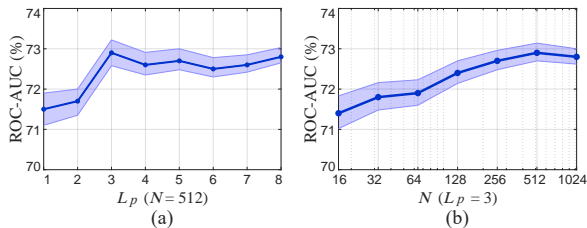


Figure 3. Sensitivity analysis of hierarchical prototypes’ depth L_p and batch size N . (All results are evaluated on biology domain.)

Sensitivity of batch size N . In this experiment, we evaluate the effect of the batch size N on our method. Fig. 3(b) shows the test ROC-AUC on downstream task using different batch sizes. From the line chart, we can observe that large batch size (*i.e.* $N \geq 256$) can promote the performance of GraphLoG. Under such condition, the sampled mini-batches can better represent the whole data set and thus derive more precise likelihood expectation in Eq. 14.

Visualization. In Fig. 2, we utilize the t-SNE (Maaten & Hinton, 2008) to visualize the graph embeddings and hierarchical prototypes on ZINC15 data set. Compared with only using the local constraints \mathcal{L}_{sub} and \mathcal{L}_{graph} (configurations (a) and (b)), more obvious feature separation is achieved after applying the global constraint \mathcal{L}_{global} (configuration (c)), which illustrates its effectiveness on discovering the underlying global-semantic structure of the data.

6. Conclusions and Future Work

We design a unified framework called Local-instance and Global-semantic Learning (GraphLoG) for self-supervised graph representation learning, which models the structure of a set of unlabeled graphs both locally and globally. In this framework, we novelly propose to learn hierarchical prototypes upon graph embeddings to infer the global-semantic structure in graphs. Using the benchmark data sets from both chemistry and biology domains, we empirically verify our method’s superior performance over state-of-the-art approaches on different GNN architectures.

Our future works will include further improving the global structure learning technique, unifying pre-training and fine-tuning, and extending our framework to other domains such as sociology, physics and material science.

Acknowledgements

This project was supported by the Natural Sciences and Engineering Research Council (NSERC) Discovery Grant, the Canada CIFAR AI Chair Program, collaboration grants between Microsoft Research and Mila, Samsung Electronics Co., Ltd., Amazon Faculty Research Award, Tencent AI Lab Rhino-Bird Gift Fund and a NRC Collaborative R&D Project (AI4D-CORE-08). This project was also partially funded by IVADO Fundamental Research Project grant PRF2019-3583139727. Bingbing Ni is supported by National Science Foundation of China (U20B2072, 61976137).

The authors would also like to thank Meng Qu, Shengchao Liu, Zhaocheng Zhu and Zuobai Zhang for providing constructive advices during this project, and also appreciate the Student Innovation Center of SJTU for providing GPUs.

References

- Alvarez, M. A. and Yan, C. A new protein graph model for function prediction. *Computational Biology and Chemistry*, 37:6–10, 2012.
- Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., et al. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.
- Bach, F. R. and Harchaoui, Z. DIFFRAC: a discriminative and flexible framework for clustering. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T. (eds.), *Advances in Neural Information Processing Systems*, 2007.
- Belkin, M. and Niyogi, P. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems*, 2002.
- Cao, S., Lu, W., and Xu, Q. Grarep: Learning graph representations with global structural information. In *ACM International Conference on Information and Knowledge Management*, 2015.
- Cappé, O. and Moulines, E. On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):593–613, 2009.
- Caron, M., Bojanowski, P., Joulin, A., and Douze, M. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018.
- Celeux, G. and Govaert, G. A classification em algorithm for clustering and two stochastic versions. *Computational Statistics & Data Analysis*, 14(3):315–332, 1992.
- Chen, B., Sheridan, R. P., Hornak, V., and Voigt, J. H. Comparison of random forest and pipeline pilot naive bayes in prospective qsar predictions. *Journal of Chemical Information and Modeling*, 52(3):792–803, 2012a.
- Chen, L., Zeng, W.-M., Cai, Y.-D., Feng, K.-Y., and Chou, K.-C. Predicting anatomical therapeutic chemical (atc) classification of drugs by integrating chemical-chemical interactions and similarities. *PloS one*, 7(4):e35254, 2012b.
- Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. E. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020.
- Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.
- Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.
- Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, 2015.
- Furnas, G. W., Deerwester, S., Durnais, S. T., Landauer, T. K., Harshman, R. A., Streeter, L. A., and Lochbaum, K. E. Information retrieval using a singular value decomposition model of latent semantic structure. In *ACM SIGIR Forum*, 2017.
- García-Durán, A. and Niepert, M. Learning graph representations with embedding propagation. In *Advances in Neural Information Processing Systems*, 2017.
- Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, 2017.
- Grover, A. and Leskovec, J. node2vec: Scalable feature learning for networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.
- Gutmann, M. and Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *International Conference on Artificial Intelligence and Statistics*, 2010.
- Gutmann, M. and Hyvärinen, A. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13:307–361, 2012.

- Hamilton, W. L., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 2017.
- Hassani, K. and Ahmadi, A. H. K. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning*, 2020.
- Hathaway, R. J. Another interpretation of the em algorithm for mixture distributions. *Statistics & Probability Letters*, 4(2):53–56, 1986.
- He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. B. Momentum contrast for unsupervised visual representation learning. *CoRR*, abs/1911.05722, 2019.
- Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V. S., and Leskovec, J. Pre-training graph neural networks. *CoRR*, abs/1905.12265, 2019.
- Hu, Z., Dong, Y., Wang, K., Chang, K.-W., and Sun, Y. Gpt-gnn: Generative pre-training of graph neural networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020.
- Ji, X., Vedaldi, A., and Henriques, J. F. Invariant information clustering for unsupervised image classification and segmentation. In *International Conference on Computer Vision*, 2019.
- Jiang, B., Kloster, K., Gleich, D. F., and Gribskov, M. Ap-trank: an adaptive pagerank model for protein function prediction on bi-relational graphs. *Bioinformatics*, 33(12):1829–1836, 2017.
- Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.
- Kipf, T. N. and Welling, M. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016.
- Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.
- Krishnan, G. J., Ng, T., Ng, S., Krishnan, T., and McLachlan, G. The em algorithm. In *Wiley Series in Probability and Statistics: Applied Probability and Statistics*, Wiley-Interscience, 1997.
- Li, J., Zhou, P., Xiong, C., Socher, R., and Hoi, S. C. H. Prototypical contrastive learning of unsupervised representations. *CoRR*, abs/2005.04966, 2020.
- Liang, P. and Klein, D. Online em for unsupervised models. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2009.
- Lu, Y., Jiang, X., Fang, Y., and Shi, C. Learning to pre-train graph neural networks. In *AAAI Conference on Artificial Intelligence*, 2021.
- Maaten, L. V. D. and Hinton, G. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2605):2579–2605, 2008.
- Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y., and Jaiswal, S. graph2vec: Learning distributed representations of graphs. *CoRR*, abs/1707.05005, 2017.
- Navarin, N., Tran, D. V., and Sperduti, A. Pre-training graph neural networks with kernels. *CoRR*, abs/1811.06930, 2018.
- Neal, R. M. and Hinton, G. E. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*. Springer, 1998.
- Nielsen, S. F. et al. The stochastic em algorithm: Estimation and asymptotic results. *Bernoulli*, 6(3):457–489, 2000.
- Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: online learning of social representations. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014.
- Qiu, J., Chen, Q., Dong, Y., Zhang, J., Yang, H., Ding, M., Wang, K., and Tang, J. GCC: graph contrastive coding for graph neural network pre-training. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020.
- Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding by generative pre-training. 2018. URL https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf.
- Rong, Y., Bian, Y., Xu, T., Xie, W., Wei, Y., Huang, W., and Huang, J. Self-supervised graph transformer on large-scale molecular data. *Advances in Neural Information Processing Systems*, 2020.
- Roweis, S. T. and Saul, L. K. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.
- Sato, M.-A. and Ishii, S. On-line em algorithm for the normalized gaussian network. *Neural Computation*, 12(2):407–432, 2000.
- Shi, C., Xu, M., Guo, H., Zhang, M., and Tang, J. A graph to graphs framework for retrosynthesis prediction. In *ICML*, pp. 8818–8827, 2020.

- Sterling, T. and Irwin, J. J. Zinc 15–ligand discovery for everyone. *Journal of Chemical Information and Modeling*, 55(11):2324–2337, 2015.
- Sun, F., Hoffmann, J., and Tang, J. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *CoRR*, abs/1908.01000, 2019.
- Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. LINE: large-scale information network embedding. In *International Conference on World Wide Web*, 2015.
- Tenenbaum, J. B., De Silva, V., and Langford, J. C. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations*, 2018.
- Velickovic, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., and Hjelm, R. D. Deep graph infomax. In *International Conference on Learning Representations*, 2019.
- Wang, D., Cui, P., and Zhu, W. Structural deep network embedding. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.
- Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.
- Xie, J., Girshick, R. B., and Farhadi, A. Unsupervised deep embedding for clustering analysis. In *International Conference on Machine Learning*, 2016.
- Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.
- Yang, B., Fu, X., Sidiropoulos, N. D., and Hong, M. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *International Conference on Machine Learning*, 2017.
- Yang, J., Parikh, D., and Batra, D. Joint unsupervised learning of deep representations and image clusters. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W. L., and Leskovec, J. Hierarchical graph representation learning with differentiable pooling. In *Advances in Neural Information Processing Systems*, 2018.
- You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., and Shen, Y. Graph contrastive learning with augmentations. In *Advances in Neural Information Processing Systems*, 2020a.
- You, Y., Chen, T., Wang, Z., and Shen, Y. When does self-supervision help graph convolutional networks? In *International Conference on Machine Learning*, 2020b.
- Zhang, G., He, H., and Katabi, D. Circuit-gnn: Graph neural networks for distributed circuit design. In *International Conference on Machine Learning*, 2019.
- Zhang, M., Cui, Z., Neumann, M., and Chen, Y. An end-to-end deep learning architecture for graph classification. In *AAAI Conference on Artificial Intelligence*, 2018.